Samuli Ristimäki

Ajanvarausjärjestelmän mobiilisuunnittelu

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikan koulutusohjelma

Insinöörityö

23.11.2015

Metropolia

Tekijä Otsikko Sivumäärä Aika	Samuli Ristimäki Ajanvarausjärjestelmän mobiilisuunnittelu 28 sivua + 3 liitettä 23.11.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikan koulutusohjelma
Suuntautumisvaihtoehto	Ohjelmistotekniikka
Ohjaajat	Projektipäällikkö Olli Venemies Yliopettaja Jaana Holvikivi

Insinöörityön tavoite oli suunnitella ja toteuttaa selainkäyttöisen ajanvarausjärjestelmän mobiiliversio terapeuttien tietojärjestelmään. Mobiiliversion tarkoituksena on helpottaa ajanvarauksien tekemistä ja ylläpitämistä erityisesti tablet-laitteilla. Haasteena oli löytää sopivat alustariippumattomat tekniikat mobiililaitteiden verkkoselaimia varten. Kehittäjän kannalta keskeinen ongelma oli näyttöresoluutioiden vaihtelevuus laitekohtaisesti.

Suunnitteluvaiheessa luotiin asiakaskysely. Kyselyn tavoitteena oli hankkia tietoa käyttäjiltä, minkä avulla voitiin kehittää käyttäjäystävällisempi järjestelmä ottaen huomioon asiakkaiden omakohtaiset käyttökokemukset. Kyselystä selvisi, että suurin osa käyttäjistä toivoi parannuksia ajanvarausjärjestelmän käytettävyyteen tablet-laitteilla.

Työhön kuului responsiivisen prototyypin kehittäminen ajanvarausjärjestelmästä. Prototyypin pääsisältönä olivat ajanvarauskalenterin päivä- ja viikkonäkymä sekä niihin kuuluvat lomakedialogit. Työssä käytetyt sovellustekniikat olivat HTML5, CSS3, PHP ja JavaScript sekä näihin ohjelmointikieliin tehtyjä kolmannen osapuolen apukirjastoja ja viitekehyksiä.

Tuloksena syntyi monipuolinen suunnitelma ja sen pohjalta tehty responsiivinen ja helppokäyttöinen prototyyppi ajanvarausjärjestelmän mobiilinäkymästä. Työ edistää järjestelmän jatkokehittämistä mobiililaitteille.

Avainsanat	responsiivisuus, web, ajanvarausjärjestelmä, mobiili

Author Title Number of Pages Date	Samuli Ristimäki Designing a mobile scheduling system 28 pages + 3 appendices 23 October 2015
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Software Engineering
Instructors	Olli Venemies, Project Manager Jaana Holvikivi, Principal Lecturer

The goal of this thesis was to plan and execute a mobile version of a scheduling system meant for therapists. The aim of the mobile version was to make scheduling easier on mobile devices, especially tablets. The challenge was to find the appropriate techniques meant for mobile device web browsers that were not tied to any certain platform. An initial issue when creating the mobile version was to take notice of the diversity of screen resolutions depending on the mobile device being used.

A user poll was sent out before execution to gather information on the users' personal experiences with the scheduling system to help create a more user friendly mobile version of the system. According to the poll, most users wish for improvements on the tablet-use of the scheduling system.

A major part of the execution was to create a responsive prototype of the scheduling system. The main focus of the prototype was on the day and week views of the scheduling calendar and their form dialogues. The techniques used to make the prototype were HTML5, CSS3, PHP, JavaScript and several third party libraries and frameworks made for these specific programming languages.

As a result, a versatile plan was created, and based on it, a responsive and easy-to-use prototype of the mobile version of a scheduling system. This thesis will help further development of the software for mobile devices.

Keywords	responsiveness, web, scheduling, mobile

Sisällys

Lyhenteet

HTML	Hypertext Markup Language. Merkintäkieli, jolla kuvaillaan www-sivujen rakenne.
PHP	Hypertext Preprocessor. Palvelinympäristössä käytettävä ohjelmointikieli.
CSS	Cascading Style Sheets. Www-sivujen tyylien määrittelemiseen käytetty tyyliohjekieli.
MVC	Model View Controller. Ohjelmistoarkkitehtuurityyli.
W3C	World Wide Web Consortium. Kansainvälinen yritysten ja yhteisöjen yhteenliittymä, joka kehittää www-standardeja.
DOM	Document Object Model. W3C:n standardoima ohjelmointirajapinta.
WAP	Wireless Application Protocol. Langattoman sovelluksen protokolla.

1 Johdanto

Diarium on Finnish Net Solutions Oy:n kehittämä terapeuteille, esimerkiksi fysioterapeuteille, toimintaterapeuteille ja hierojille, suunniteltu potilaskortisto- ja laskutusohjelma, ja sitä voi käyttää eri päätelaitteilla. Ohjelma on selainkäyttöinen, ja se sisältää useita terapeutin arkea helpottavia ominaisuuksia, kuten ajanvaraus, asiakasrekisteri, raportointi ja asiakkaan laskuttaminen. Se toimii kaikilla päätelaitteilla (tietokone, älypuhelin, tablet-laite). Ohjelman nykyinen sähköinen ajanvarauskalenteri toimii kuitenkin mobiililaitteilla heikosti. Ajanvaraus on yksi ohjelman keskeisimpiä toimintoja, ja sen mobiilitoimivuutta halutaan parantaa. Asiakkailta saadun palautteen perusteella kalenterin käyttö on hankalaa ja hidasta erityisesti tablet-laitteilla.

Opinnäytetyön tavoitteena on tehdä hyvä ja toteutuskelpoinen suunnitelma Diarium-ohjelman ajanvarauskalenterin mobiiliversion toteuttamiseksi. Lisäksi opinnäytetyön yhteydessä toteutetaan prototyyppi, jonka kautta suunnitelman toimivuus voidaan todentaa.

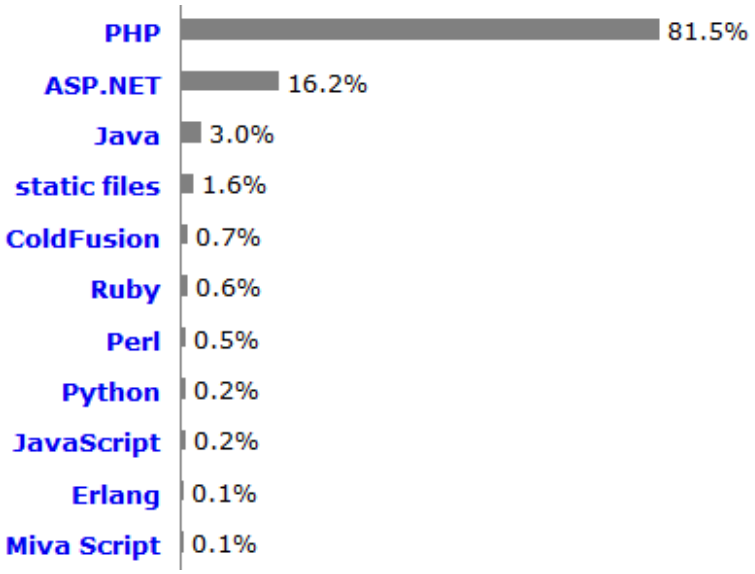
Finnish Net Solutions (FNS) on vuonna 2001 perustettu ohjelmistoyritys, joka työllistää 30 ohjelmistoalan ammattilaista Espoon, Lohjan ja Kuopion toimistoissa. Yrityksen sovelluksia hyödynnetään eläinlääkinnässä, fysioterapia-alalla ja useilla muilla erikoistoimialoilla. Kehitykseen käytetään ketterän kehityksen menetelmiä, ja työskentely tapahtuu yhteistyössä asiakkaiden kanssa. [1.]

Diarium on selainkäyttöinen verkkopalvelu, joka on asennettu FNS:n palvelinjärjestelmään. Palvelinlaitteet ovat yrityksen omistuksessa, ja ne on asennettu ulkopuoliselta paveluntarjoajalta vuokrattuun palvelintilaan. Diariumin avulla voi ylläpitää asiakasrekisteriä, tehdä hoitokirjauksia ja laskuttaa asiakasta helposti ja nopeasti. Ohjelman käyttämiseen tarvitaan ainoastaan WWW-selain ja verkkoyhteys. Diarium-ohjelmaa käytetään tietokoneilla, tablet-laitteilla ja älypuhelimilla. Mobiilikäyttö on asiakkaille hyvin tärkeä ominaisuus. Kaikki asiakkaat käyttävät samaa Diarium-ohjelman versiota. Päivitykset asennetaan keskitetysti kaikille asiakkaille FNS:n toimesta. Päivitykset ja varmuuskopiointi kuuluvat ylläpitopalveluun, ja ne tapahtuvat automaattisesti. Diarium-ohjelman käyttäjät ovat pääasiassa fysioterapeutteja ja toimintaterapeutteja. Käyttäjien joukossa on myös puheterapeutteja, jalkaterapeutteja, psykoterapeutteja ja hierojia. [2.]

2 Insinöörityössä käytetyt tekniikat

2.1 PHP-ohjelmointikieli

PHP:n eli Hypertext Preprocessorin ensimmäinen versio on julkaistu vuonna 1994, ja uusin versio 5.6.14 julkaistiin lokakuussa 2015. PHP on palvelinpohjainen ohjelmointikieli, jota käytetään dynaamisten verkkosivujen luonnissa, mikä tarkoittaa, että verkkosivu reagoi käyttäjän antamiin syötteisiin tai esimerkiksi ajankohtaan. Palvelinpohjaisuudella tarkoitetaan sitä, että PHP on komentosarjakieli, jossa koodi tulkitaan vasta ohjelman suoritusvaiheessa palvelimella. Palvelinpohjaisuuden takia PHP-skriptillä on pääsy palvelimen tiedostoihin ja tietokantoihin, joihin selaimella ei pääse. PHP:ta on käytetty 81,5 %:ssa kaikista palvelinpohjaisista verkkosivuista, joiden lähdekoodi on tiedossa (kuva 1). [3.]



Kuva 1. Tilasto palvelinpohjaisten ohjelmointikielien esiintyvyydestä verkkosivuilla marraskuussa 2015 [4].

PHP itsessään sisältää suuren määrän erilaisia funktioita, esimerkiksi tietokantayhteyden muodostamiseen ja sähköpostin lähettämiseen. Myös käyttäjäyhteisö on tehnyt useita funktioita, joita löytyy julkisesti internetistä. PHP:n funktioiden ja palvelinpohjaisuuden avulla voidaan siis luoda esimerkiksi salasanasuojaus verkkosivulle (esimerkkikoodi 1).

```
<?php
$kayttajatunnus = $_POST['kayttajatunnus'];
$salasana = $_POST['salasana'];

$yhteys = mysql_connect("localhost", "root", "");
$tietokanta = mysql_select_db("yritys", $yhteys);
$kysely = mysql_query("SELECT * FROM kayttajat WHERE salasa-
na='$salasana' AND kayttajatunnut='$kayttajatunnus'", $yhteys);

$riveja = mysql_num_rows($kysely);
if ($riveja == 1) {
            header("location: profiili.php");
} else {
            $virhe = "Käyttäjätunnus tai salasana on väärin.";
}

mysql_close($yhteys);
?>
```

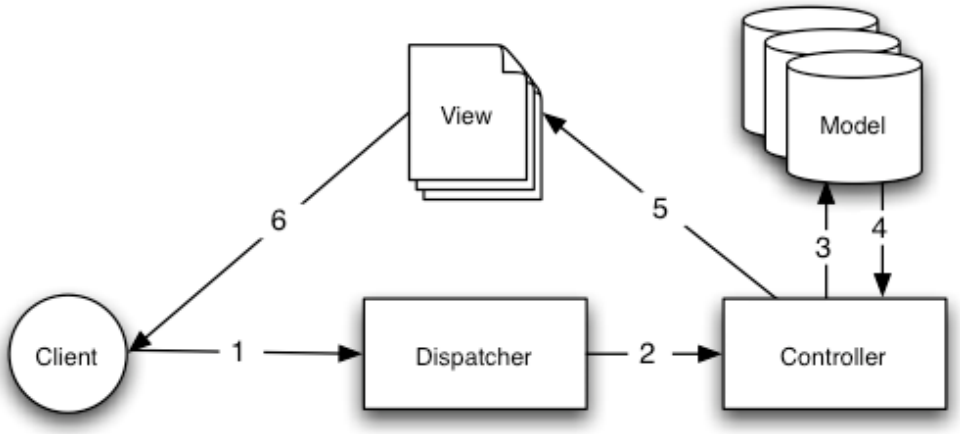
Esimerkkikoodi 1. PHP-salasanasuojaus.

2.2 CakePHP-ohjelmointikehys

CakePHP on avoimeen lähdekoodiin perustuva ilmainen verkkosovellusten MVC (Model-View-Controller, suom. malli-näkymä-ohjain) viitekehys. Se on suunniteltu tekemään PHP:n web-kehityksestä helpompaa ja tehokkaampaa esimerkiksi tarjoamalla valmiita komponentteja kehitysympäristöön. CakePHP:lla on aktiivinen kehitysyhteisö, mikä takaa sen, että sovelluksen ydin on virallisesti testattu ja sitä parannetaan jatkuvasti. [5, s. 1.]

CakePHP-viitekehys julkaistiin ensimmäisen kerran MIT-lisenssin alaisena joulukuussa 2005. Maaliskuussa vuonna 2015 julkaistiin CakePHP:n versio 3.0.0. Päivityksessä parannettiin reittien nopeuksia, lokalisaatiokirjastoa ja testaustyökalua. [6.]

Kuvassa 2 on mallinnus CakePHP:n toiminnasta esimerkiksi verkkosivua ladattaessa. Tyypillinen CakePHP-pyyntösykli alkaa, kun käyttäjä pyytää sivua tai resurssia sovelluksessa. Tämä pyyntö prosessoidaan ensin lähettäjässä (dispatcher), joka valitsee oikean ohjaimen (controller) käsittelemään sitä. Kun pyyntö saapuu ohjaimeen, se on yhteydessä malli-kerrokseen (model) tietoja noudettaessa tai tallennettaessa. Tämän jälkeen viestintä on ohi. Ohjain delegoi näkymälle (view) tehtäväksi generoida tulosteen mallin tarjoamasta datasta. Lopuksi, kun tämä tuloste on generoitu, se tarjotaan välittömästi käyttäjälle (client). [7.]



Kuva 2. Tyypillinen MVC pyyntö CakePHP:ssä [6].

2.3 JavaScript-komentosarjakieli

JavaScript on ohjelmointikieli, jota yleensä käytetään dynaamisten selainskriptien tekemiseen. JavaScript soveltuu interaktiivisten eli vuorovaikutteisten toimintojen luomiseen www-sivuille. Ohjelma voi olla hyvinkin yksinkertainen esimerkiksi ponnahdusikkuna, jonka käyttäjä voi kuitata klikkaamalla ikkunan OK-painiketta. [8, s. 8.]

JavaScript julkaistiin ensimmäisen kerran vuonna 1995 Netscape Navigator verkkoselaimen mukana. JavaScript oli silloin nimetty Livescriptiksi, mutta nimettiin nopeasti JavaScriptiksi, sillä Netscape oli yhteistyösopimuksessa Sun Microsystemsin kanssa, joka omisti Javan. Vaikka Javalla ja JavaScriptillä ei ole paljoa yhteistä, haluttiin sen nimen viittaavan Javaan, joka oli jo käytössä laajalti. [8, s. 7.]

ECMAScript on skriptauskieli, joka muodostaa Javascriptin perustan. ECMAScriptin on standardoinut Ecma International-organisaatio. Nykyinen versio on ECMA-262 2015 (Edition 6), joka julkaistiin kesäkuussa vuonna 2015. [9.]

JavaScript on alun perin suunniteltu parantamaan verkkosivuja, mutta sitä käytetään nykyään lähes kaikin tavoin web-kehityksessä. On edistytty siihen pisteeseen, että JavaScript toimii esimerkiksi palvelinpuolella ja natiivissa puhelinsovelluskoodissa. JavaScriptille on kehitetty satoja erilaisia ohjelmointiympäristöjä, viitekehyksiä ja työkaluja. [10.]

2.4 jQuery-kirjasto

jQuery on ilmainen avoimen lähdekoodin Javascript-kirjasto, joka on suunniteltu muun muassa tekemään DOM-elementtien (Document Object Model) hallinnasta ja animoinnista mahdollisimman helppoa ja selainystävällistä. DOM-elementti voi olla esimerkiksi DIV-, HTML- tai BODY-elementti verkkosivulla. jQueryn avulla voidaan yksinkertaistaa HTML-dokumentin käsittelyä ja animointia sekä tuottaa sivuille dynaamisia käyttöliittymäkomponentteja. jQueryssä on otettu huomioon erilaiset Javascript-toteutukset kaikissa yleisimmissä selaimissa, ja tuki on taattu vähintään selaimen uusimmalle ja toiseksi uusimmalle versiolle. jQuery yksinkertaistaa JavaScriptin käyttöä ja erityisesti mahdollistaa yhtenevän toteutuksen niin uusille kuin vanhoille selaimille. [8, s. 23.]

jQuerysta on kaksi versiota (1.x ja 2.x). jQuery 2.x käyttää samaa ohjelmointirajapintaa kuin jQuery 1.x, mutta ei ole yhteensopiva Internet Explorerin versioiden 6, 7 tai 8 kanssa. Uusin julkaistu päivitys versiosta 2.x on 2.1.4. jQuerya kehittää yhteisö nimeltä The jQuery Foundation. Yhteisö on jäsenten tukema voittoa tavoittelematon web-kehittäjien yhdistys. [11.]

Verkkosovelluksien kehittäjät hyötyvät jQuery-kirjaston yhteensopivuudesta, koska se toimii kaikissa moderneissa verkkoselaimissa. jQuery on suosituin JavaScript-kirjasto, eli sen käyttö on hyvin yleistä. Yli 53 miljoonaa www-sivustoa käyttää jQuery-kirjastoa. [12.]

2.5 Bootstrap-työkaluvalikoima

Bootstrap on yleiskäyttöinen työkaluvalikoima, jolla voi nopeasti toteuttaa modernin käyttöliittymän. Alun perin vuonna 2010 Bootstrapin kehitti Twitterin kaksi työntekijää, Mark Otto ja Jacob Thorton. Se tarjoaa valmiita käyttöliittymän rakennuspalikoita, kuten ruudukkojärjestelmiä, navigointiin tarvittavia komponentteja, dialogeja ja lomake-elementtejä. Kehys auttaa yhtenäistämään selainten oletusarvoisia tapoja muotoilla HTML-elementtejä. Lisäksi mukana on responsiivisen käyttöliittymän toteuttamiseen tarvittavia määrityksiä. [13.]

Bootstrapilla säästää paljon aikaa ja vaivaa käyttämällä valmiita suunnittelumalleja ja luokkia. Sillä voi helposti luoda responsiivisia näkymiä, joiden reagoivat ominaisuudet tekevät verkkosivun näkymän toimivammaksi eri laitteilla ja näyttöresoluutioilla. Kaikki Bootstrap-komponentit jakavat samoja suunnittelumalleja ja tyylejä, jotta mallit ovat verkkosivulla yhdenmukaisia koko kehityksen ajan. Bootstrap on erittäin helppokäyttöinen. Kuka tahansa, joka osaa HTML:n ja CSS:n perusteet, voi oppia Bootstrapin käytön. Bootstrapia kehitetään uudet selaimet mielessä pitäen, ja se on yhteensopiva kaikkien uusien selaimien kanssa, kuten Mozilla Firefox, Google Chrome, Safari, Internet Explorer ja Opera. Bootstrap perustuu avoimeen lähdekoodiin, joten se on ilmainen ladata ja käyttää. [13.]

Bootstrapin hyödyt tulevat esille, jos tavoitteena on luoda käytännöllinen ja moderni verkkosivu nopeasti. Ruudukkojärjestelmä, apuluokat ja valmiit tyylimäärittelyt tarjoavat laajat lähtökohdat ilman, että kehittäjän tarvitsee kuluttaa aikaa niiden kirjoittamiseen. Haittapuolena ovat mahdolliset konfliktit kehittäjän omien tyylimäärittelyjen kanssa. Kaikki elementit on ennalta määritelty tietynlaisiksi, ja ne yliajavat oletuksena muut tyylimäärittelyt. Tämän estämiseksi CSS-koodissa voidaan käyttää !important-tagia, joka priorisoi tyylimäärittelyn tärkeimmäksi ja yliajaa muut saman elementin vastaavat tyylit.

3 Mobiilisuunnittelu

3.1 Ketterä ohjelmistokehitys

Ketterä ohjelmistokehitys on vaihtoehto perinteiselle projektinhallinnalle. Sen tarkoituksena on jakaa toteutus jaksoihin, joiden avulla varmistetaan, että ohjelmisto vastaa asiakkaan tarpeita. Ketterän kehityksen menetelmiin kuuluvan suoran viestinnän avulla asiakkaiden muuttuviin toiveisiin pystytään reagoimaan nopeasti projektin aikana. Tämä säästää resursseja molemmilta osapuolilta. [14.]

Vuonna 2001 merkittäviä ketterän kehityksen puolestapuhujia kokoontui keskustelemaan menetelmiensä yhteisestä perustasta. Tarkoitus oli luoda yhteistä pohjaa ketterille menetelmille ja edistää näin ketterän ajattelun leviämistä. Tuon kokoontumisen tuloksena julkaistiin julistus nimeltä ”Ketterä manifesti”, jota pidetään ketterän kehityksen perusmääritelmänä. Manifestissa määritellään ketterille menetelmille neljä tyypillistä arvoa, joita ne noudattavat. [15.]

Ketterän ohjelmistokehityksen julistuksen arvot:

- **Yksilöitä ja vuorovaikutusta**: Henkilökohtaisen järjestäytyneisyyden ja motivaation ylläpitäminen on yhtä tärkeää kuin työntekijöiden keskeinen vuorovaikutus, esimerkiksi pariohjelmointi.
- **Toimivaa sovellusta**: Toimivan sovelluksen luominen käytännössä on tärkeämpää kuin dokumentaation esittely asiakkaille.
- **Asiakasyhteistyötä**: Jatkuva asiakaskysely ja -tutkimus ovat edellytys tehokkaalle ohjelmistokehitykselle.
- **Muutokseen reagoimista:** Ketterä kehitys keskittyy nopeaan reagoimiseen muutoksiin ja jatkuvaan kehitykseen.

[16.]

3.2 MVC-arkkitehtuuri

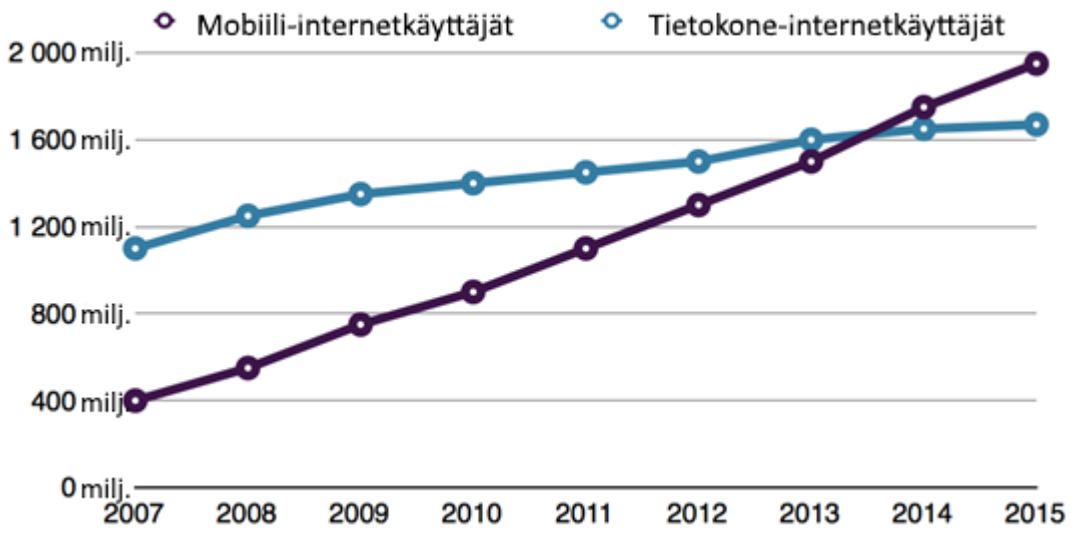
MVC (Model-View-Controller, suom. malli-näkymä-ohjain) on ohjelmointiarkkitehtuuri, jonka keskeinen ajatusmalli on jakaa ohjelma helposti hallittaviin loogisiin osiin: malliin, näkymään ja ohjaimeen. Tämän tarkoituksena on helpottaa ohjelman muokkaamista ja

tutkimista päällekkäisyyksiä minimoimalla (separation of concerns). Tämä osiin jakaminen helpottaa erityisesti monimutkaisten ohjelmien hallittavuutta ja kehittämistä, sillä eri osia voi tarkastella ja kehittää erillään toisistaan. MVC:stä on hyötyä etenkin tiheään tahtiin kehitystä vaativien sovellusten ja sivustojen kehityksessä, sillä muokkaaminen on nopeaa osien erittelyn vuoksi. [16.]

MVC on hyödyksi, kun rakennetaan modernia verkkosovellusta, joka sisältää dynaamisia osia ja komponentteja. MVC nopeuttaa sovelluksen kehityskulkua, koska sen avulla koodista tulee uudelleenkäytettävää. MVC-arkkitehtuuri mahdollistaa, että näkymiä päivitetään, kun niihin liittyvän mallin data muuttuu. Sivua ei tarvitse ladata kokonaan uudelleen, vaan voidaan päivittää vain tarvittavat näkymät. [16.]

3.3 Responsiivisuus

Responsiivinen suunnittelu tai mukautuva suunnittelu on yksi ajankohtaisimmista websuunnitteluun liittyvistä puheenaiheista. HTML5:n ja CSS3:n julkaisemisen jälkeen verkkoprojektien yleinen vaatimus on, että näkymän tulisi toimia myös mobiililaitteilla. Tämä johtuu siitä, että mobiililaitteiden käyttö on nopeasti lisääntynyt tällä vuosikymmenellä. [17, s. 23.]



Kuva 3. Mobiililaitteiden internetkäytön lisääntyminen [18].

Perinteisesti mobiilisivuja suunnitellessa on käytetty kahta eri tapaa. Ensimmäinen niistä on, että luodaan erillinen web-mobiilisivusto mobiililaitteita varten. Ylläpidettävänä

tällöin on kaksi sivustoa, jotka eivät sisällä yhteistä lähdekoodia. Toinen tapa on luoda erillinen natiivisovellus. Tällöin joudutaan tekemään ja ylläpitämään erillisiä sovelluksia. Työlääksi tämän tekee se, että kokonaisuuden muuttuessa on yksi tai useampia natiivisovelluksia päivitettävänä erikseen. Koska kumpikaan mainituista tavoista ei ole välttämättä optimaalisin, on kehitetty responsiivinen tapa toteuttaa verkkosivuja. Responsiivisuudella tarkoitetaan käyttöliittymäsuunnittelua, jonka avulla verkkosivua voidaan käyttää eri päätelaitteilla ilman, että käyttökokemus suoranaisesti kärsisi näytön koon tuomista rajoituksista. Responsiivisen suunnittelun tarkoituksena voidaan siis pitää, että voidaan luoda käytettävyydeltään toimivat näkymät erilaisiin käyttötarkoituksiin (kuva 4). [17, s. 24.]

Kuva 4. Responsiivisessa suunnittelussa käyttöliittymiä mukautetaan eri päätelaitteille [9].

4 Diarium-ajanvarausjärjestelmän mobiilisuunnittelu

4.1 Asiakaskysely

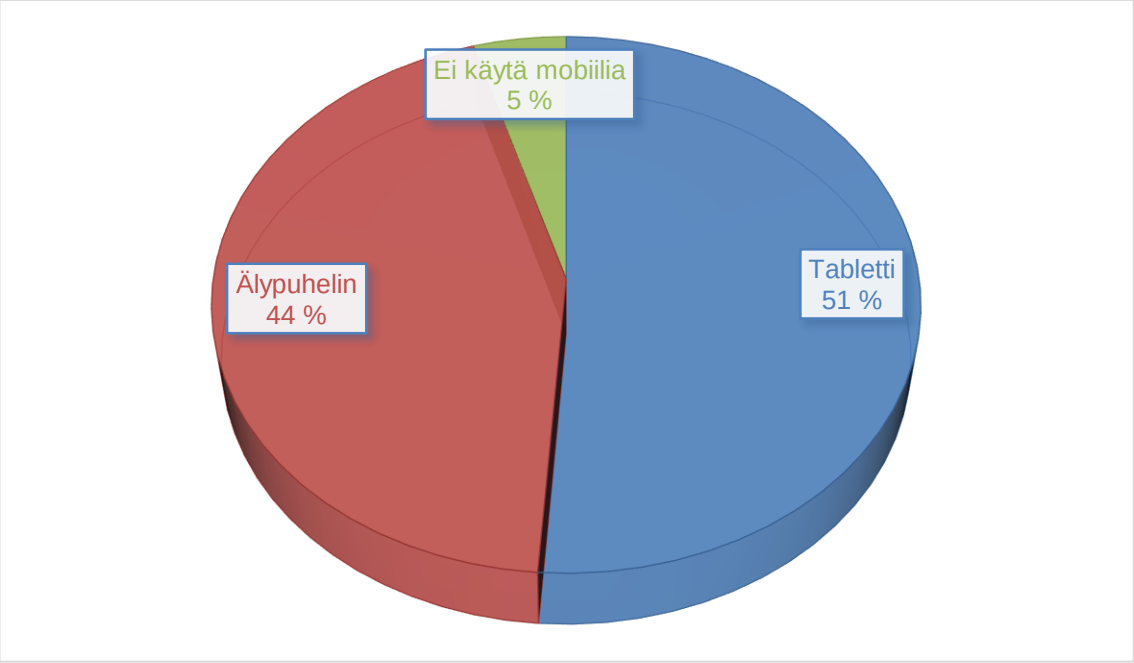
Insinöörityön suunnitteluvaiheessa toteutettiin asiakaskysely, jonka tavoitteena oli hankkia yksilöllistä tietoa suoraan Diariumin mobiilikäyttäjiltä. Tämän avulla voidaan tulevaisuudessa kehittää käyttäjäystävällisempi järjestelmä ottaen huomioon asiakkaiden käyttökokemukset. Asiakaskysely luotiin käyttäen Google Forms -palvelua. Kyse-

lyyn vastasi yhteensä 63 asiakasta. Kyselyiden tärkeys ilmenee, kun valmis ominaisuus pääsee asiakkaiden käyttöön. Asiakkaiden vaatimukset toteutetaan tärkeysjärjestyksessä, mikä minimoi valmiin ominaisuuden palautteissa negatiiviset mielipiteet.

Asiakaskyselyn kysymykset:

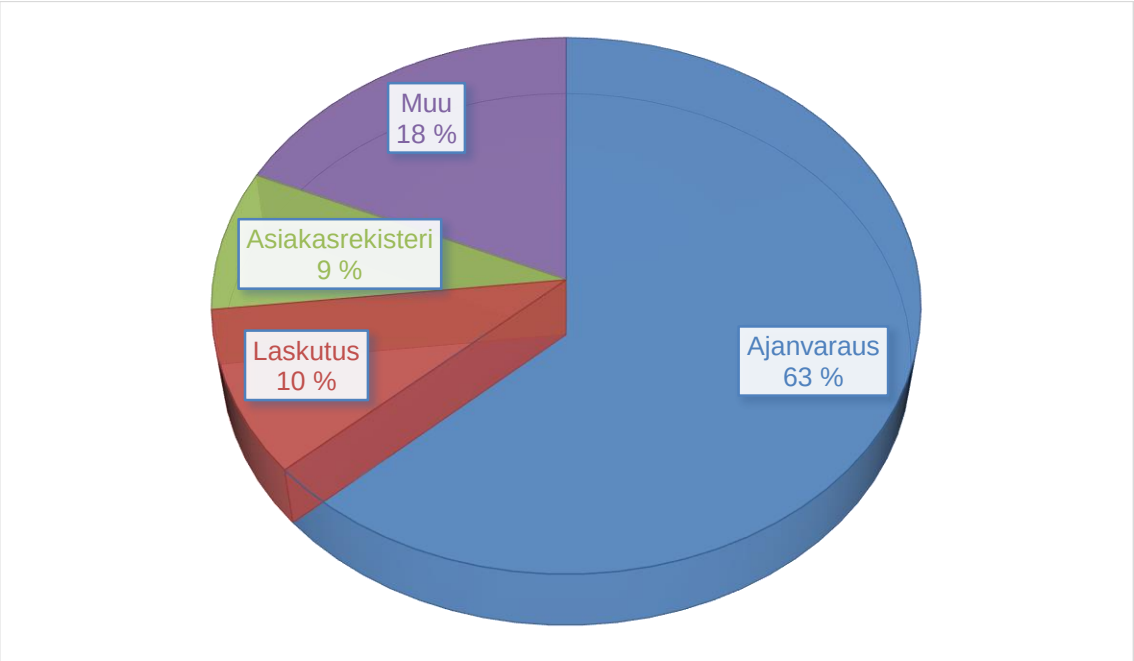
- Millä mobiililaitteella käytät Diariumia?
- Nimeä kolme tärkeintä ominaisuutta, joita käytät tai haluat käyttää mobiililaitteella.
- Mihin ominaisuuksiin erityisesti haluttaisiin parannuksia?
- Haluatko pilottitestata tulevaa Diarium-mobiiliajanvarausta?

Odotetusti suurin osa mobiilikäyttäjistä käyttää Diariumia tabletilla (kuva 5). Myös valtaosa älypuhelinkäyttäjistä käyttää Diariumia tabletilla älypuhelimen rinnalla. Kyselyn perusteella päivittäisessä työkäytössä siis suositaan suurempaa näytön kokoa.



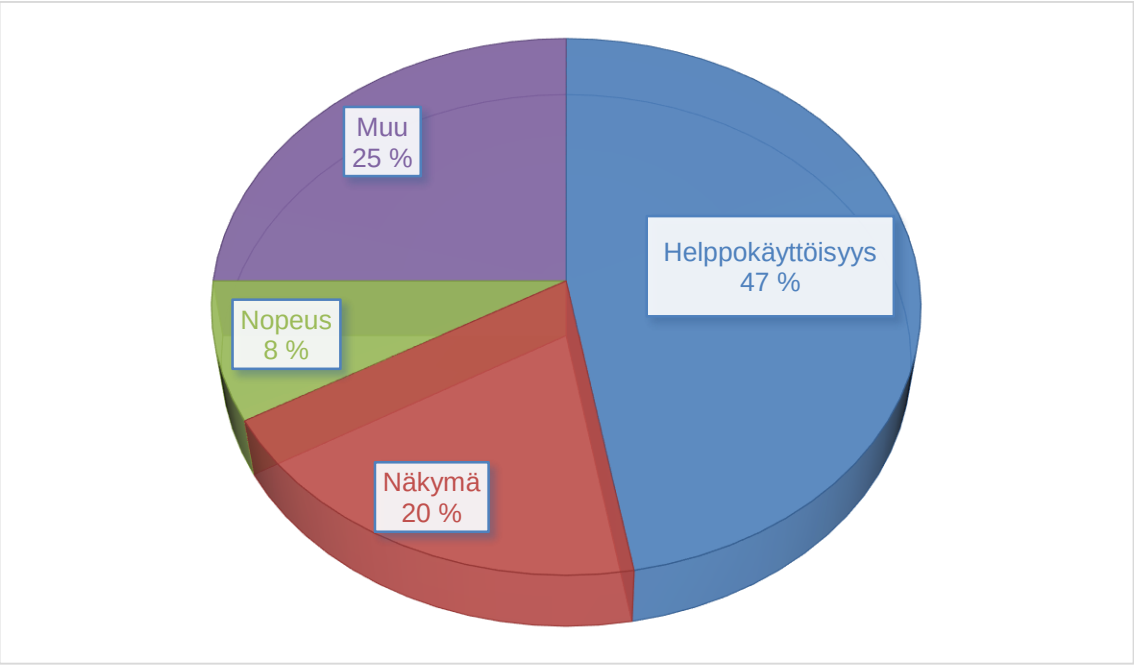
Kuva 5. Mobiililaitteet, joilla Diariumia käytetään.

Ennen kyselyä asiakaspalautteiden perusteella tiedettiin, että oli monia käyttäjiä, jotka halusivat käyttää Diariumin ajanvarausominaisuutta mobiilisti (kuva 6). Koska ajanvaurausominaisuus on niin laajalti mobiilikäytössä, tulee ottaa järjestelmäkehityksessä huomioon suuri käyttäjämäärä ja sen vaikutukset ohjelman toimintaan.



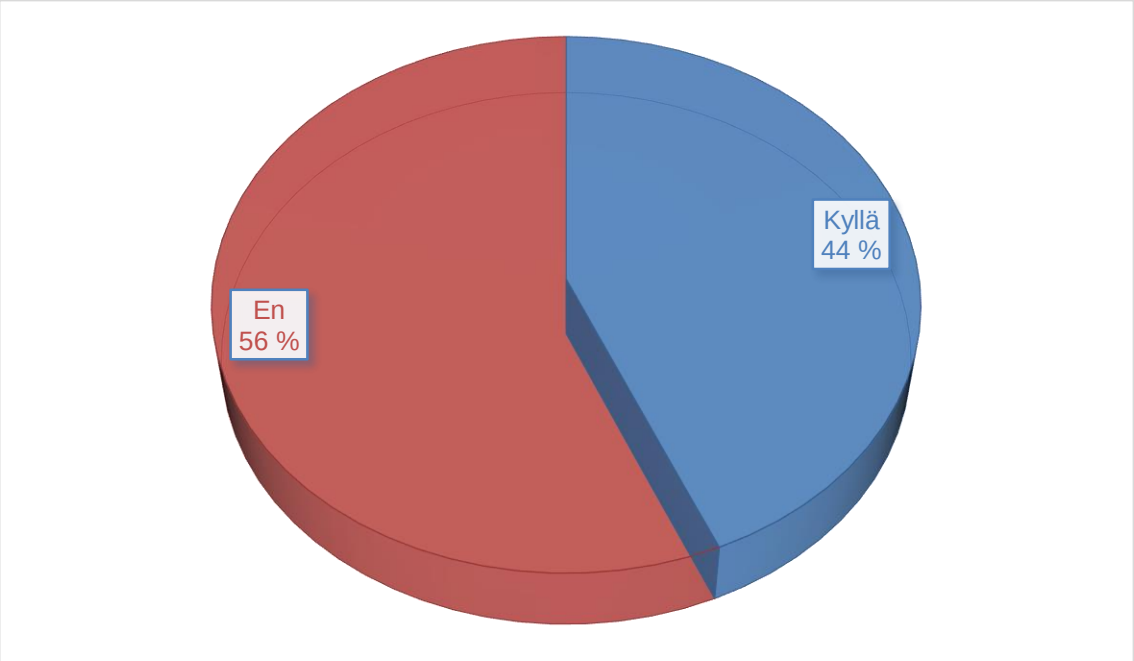
Kuva 6. Diariumin eniten käytetyt ominaisuudet mobiililaitteilla.

Nimeämällä kolme tärkeintä ominaisuutta saadaan tietoon järjestelmän parannusta vaativat ominaisuudet asiakkaan näkökulmasta, mikä on hyvin tärkeää palveluntarjoajalle. Helppokäyttöisyys on eniten toivottu mobiilisovelluksen ominaisuus (kuva 7).



Kuva 7. Diariumin käyttäjien yleisimmät kehitystoiveet mobiililaitteille.

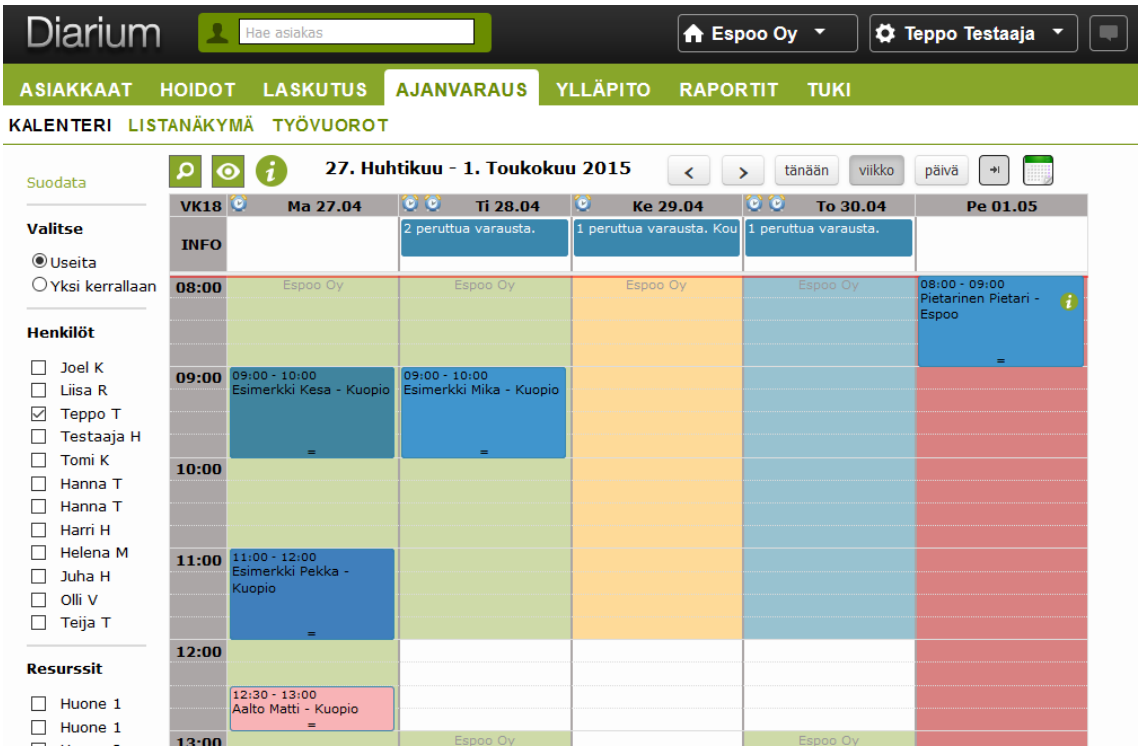
Suurin osa käyttäjistä vastasi kielteisesti pilottikokeiluun, koska he eivät välttämättä koe omaa testidataansa tarpeelliseksi (kuva 8). On kuitenkin positiivista, että riittävä määrä käyttäjistä on tarvittaessa halukas kokeilemaan pilottiversiota, sillä sen avulla saadaan tärkeää palautetta jo järjestelmän kehitysvaiheessa.



Kuva 8. Käyttäjien halukkuus Diariumin mobiilipilottiversion kokeiluun.

4.2 Suunnitelma

Lähtökohtana on luoda mobiilioptimoitu versio Diariumin ajanvaraustoiminnosta. Keskeisenä tavoitteena on erotella mobiiliversioon tarvittavat ja tarpeettomat ominaisuudet. Tämän lisäksi erityisen tärkeää on optimoida ajanvarauksessa käytettävä kalenteri- ja dialogikomponentti mobiilikäyttöön ja selvittää, mistä mobiiliversion hitaat latausnopeudet johtuvat.



Kuva 9. Diariumin alkuperäinen ajanvarausnäkymä.

Tarkoituksena on muokata ajanvarausnäkymän elementit suuremmiksi, jotta sen tarkastelu ja käyttö pienemmällä näytöllä olisi helpompaa. Painikkeiden, syöttökenttien ja linkkien suurentaminen on käyttäjäystävällinen muutos, jos käytössä olevan laitteen voidaan olettaa olevan kosketusnäytöllinen. Näytön käyttäminen sormella vaatii näkymään suuret elementit. Kalenterin tunnit halutaan ilman tuntijakoa, jotta voidaan minimoida käyttäjän virhepainallusten määrä.

Päävalikon piilottaminen selkeyttää vakionäkymää, sillä se ei mahdu järkevästi pienelle näytölle. Valikon piilottaminen tekee kalenterin käytöstä helpompaa, sillä täten näytöllä ei ole mitään ”ylimääräistä”. Tätä periaatetta varten on kehitelty monia mobiilivalikoita, jotka yleensä toimivat nappia painamalla ja avautuvat ylhäältä alaspäin listana. Kun valikkoa tarvitaan, sen saa helposti avattua, mutta muuten se pysyy poissa tieltä.

Diariumin kalenterina on käytössä JavaScript-komponentti nimeltä FullCalendar. Mobiililaitteille on saatavana useita erilaisia kalenteriratkaisuja, mutta Diariumin jo valmiina olevat ajanvarausominaisuudet on tehty FullCalendarin ympärille. FullCalendar toimii liitännäisenä hyvin mobiililaitteiden kanssa, joten mobiilinäkymässä halutaan käyttää jo valmiina olevaa kalenteria.

Ylimääräisen karsinta on mobiiliversiolle olennainen muutos, sillä näytön pieni koko rajoittaa elementtien määrää. Mobiiliversiossa pyritään ennen kaikkea selkeyteen ja helppokäyttöisyyteen, mikä vaatii mahdollisimman yksinkertaisen version ajanvaraus-järjestelmästä. Lähtökohtana on piilottaa kaikki, mikä ei ole jatkuvassa käytössä.

Vähemmän käytössä olevan sisällön jättäminen tarvittaessa kokonaan pois on vaihtoehto. Tämä saattaa vaatia sitä, että sen saa kuitenkin haluttaessa esille, esimerkiksi painikkeella. Fonttivalikoiman huomioon ottaminen on myös tärkeää. Mobiililaitteissa on usein hyvin suppea kirjaintyyppivalikoima. Joissakin on vain kirjainlajit, jotka vastaavat CSS:n geneerisiä kirjainnimiä sans-serif, serif ja monospace. Muiden kirjainlajien käyttämiseksi ne on erikseen ladattava. Koristekuvien ja vastaavien tilaa vievien elementtien jättäminen pois tarvittaessa on myös mahdollista.

Mobiilisivuissa suositaan usein koko näytön leveyden kattavuutta, jotta laitteen koko näyttö tulee hyödynnettyä, mikä jälleen mahdollistaa mahdollisimman suurena näkyvät elementit ja täten tekee sivusta helppokäyttöisemmän. Tämä saadaan aikaan CSS-tyyleillä asettamalla määrite `width` (leveys) 100 prosenttiin.

Mobiilioptimoinnissa on otettava huomioon sivuston raskaus ja latausajat. Mobiililaitteiden yhteydet ovat usein hitaampia kuin tietokoneiden. Jos kyseessä on paljon sisältöä sisältävä sivu, on hyvä, jos sivusto on selattavissa jo ennen kuin kaikki sisältö on ehtinyt latautua. Sivun latautuessa taustalla voidaan käsitellä etukäteen myös näkymän piilossa olevien elementtien tietoja.

4.3 Prototyyppi

4.3.1 Media query -tyylimääritys

Ajanvaraus.css:n (liite 1) mediakyselyjen (media query) avulla voidaan korjata epäkohtia, joita ilmenee näytön resoluution muuttuessa. Käytännössä tämä tarkoittaa, että mediakyselyillä voidaan määrittää esimerkiksi resoluutio tai leveys sen mukaan, minkäkokoinen näyttö on käytössä. Tämän ominaisuuden avulla verkkosivuista saadaan juuri sopivat kaikenkokoisille näytöille.

CSS-komennon yleismuoto on

```
@media mediatype and|not|only (media feature){

        CSS;

}
```

Yleismuodossa oleva mediatyyppi (mediatype) määrittää kohdelaitteen tyypin, esimerkiksi `screen`. Mediatyyppi komentaa käyttäjäagenttia (user agent), tässä tapauksessa selainta, lataamaan mediakyselyjen määrittämät tyylit. Kyselyjä voidaan myös yhdistää `and`-, `or`- ja `not`-operaattoreiden avulla ja täten luoda monimutkaisempia komentosarjoja.

Kysely koostuu ominaisuudesta (feature) ja sille annetusta arvosta (value). Esimerkkikoodi, jossa mediatyyppi on näyttö ja kyselylle ”maksimileveys” määritetään arvo 2 560 pikseliä:

```
@media screen and (max-width: 2560px){

        CSS;

}
```

Mediakyselyssä siis määritellään, minkäkokoisella näytöllä aaltosulkeiden sisällä olevat tyylit ladataan (esimerkkikoodi 2 ja kuva 10).

```
@media only screen
and (min-device-width : 768px)
and (max-width: 2560px) {
  #leftbar {
          display: none;
  }
  #calendar{
          width: 98%;
  }
}
```

Esimerkkikoodi 2. Mediakysely, joka piilottaa ajanvarausnäkymän vasemman työkalupalkin ja leventää kalenterin koko näytön leveydelle tablet-laitteiden näytön koossa.

Kuva 10. Diarium-ajanvarausnäkymän sivupalkin piilotus mediakyselyn avulla tablet-laitteissa.

4.3.2 Viewport-tagi

Responsiivisen ajanvarauskalenterin halutaan ensisijaisesti näkyvän verkkoselaimessa kokonaisena laitteen näytön koosta ja muodosta riippumatta. Tämä helpottaa kalenterin käyttöä ja tarkoittaa käytännössä sitä, että käyttäjä näkee heti avatessaan koko sivun yhdellä silmäyksellä, minkä jälkeen sitä voi skaalata, eli suurentaa tai pienentää, haluamansa mukaan. Ongelma verkkosivun kehittäjän näkökulmasta syntyy, kun halutaan sivun skaalautuvan kokonaisena kaikenkokoisille ja -muotoisille näytöille. Skaalaustapaa voidaan muuttaa käyttämällä HTML-kielen metatieto-ominaisuutta ja viewport-tagia. Viewport tarkoittaa suomeksi näyttöruutua eli verkkosivun ruudulla olevaa näkymää. [10.]

Viewport-tagi sijoitetaan `<head>`-tagin sisään:

```
<meta name=”viewport” content=”attr1, attr2, attr3, …”>
```

Määrite `user-scalable` määrittää, voiko käyttäjä zoomata verkkosivua. Arvo on joko `yes` tai `no`. Esimerkkinä koodi zoomauksen sallimisesta:

```
<meta name=”viewport” content=”user-scalable=yes”>
```

`Initial-scale`-määritteellä määritellään verkkosivun prosentuaalinen suurennus suhteessa käytettävän laitteen näytteen kokoon sivua avattaessa. Tyypillisesti halutaan konfiguroida verkkosivu skaalautumaan 100-prosenttiseksi. Parametrin arvo 1.0 tarkoittaa 100 %:a jne. Esimerkki:

```
<meta name=”viewport” content=”initial-scale=1.0”>
```

Määrite `width` (leveys) kertoo näyttöruudun leveyden. Arvo `device-width` tarkoittaa käytössä olevan laitteen leveyttä, joten sivu avautuu tarkalleen laitteen leveyteen sopivana. Esimerkiksi, jos ruutu on 480 pikseliä leveä, verkkosivu avautuu 480 pikselin levyisenä. Tätä käytetään tyypillisesti, sillä sivuttain vierittäminen ei ole mielekästä. Esimerkki:

```
<meta name=”viewport” content=”width=device-width”>
```

Sama pätee määritteeseen `height` (korkeus), mutta toisin kuin leveyttä määritettäessä, koodia `<meta name>=”viewport” content=”height=device-height”>` käytetään harvemmin, sillä se estäisi pystysuunnassa vierittämisen, mikä ei useimmiten ole toivottua.

```
<meta name=”viewport” content=”height=device-height”>
```

Määritteet `maximum-scale` ja `minimun-scale` määrittävät sivun suurennuksen maksimi- ja miniarvot. Parametrin arvot ovat samaa muotoa kuin `initial-scale`-määritteen arvot. Esimerkki, jossa verkkosivua ei voi suurentaa:

```
<meta name=”viewport” content=”maximum-scale=1”>
```

```
<meta name="viewport" content="initial-scale = 1.0">
```

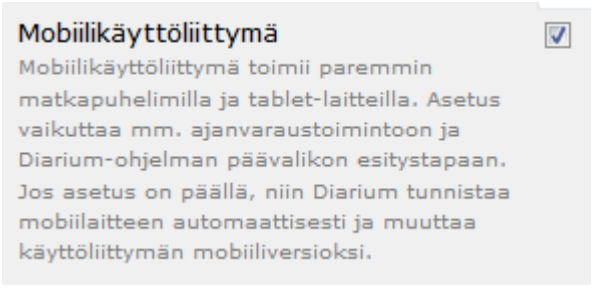
Esimerkkikoodi 3. Viewport-tagi, joka skaalaa ajanvarausnäkymän koko ruudun leveydelle.

Kuva 11. Diarium-ajanvarausnäkymän muutos viewport-elementin jälkeen tabletilla.

4.3.3 Cakephp isMobile -funktio

CakePHP sisältää isMobile-funktion, jolla pystytään tarkastamaan, onko päätelaite mobiililaite. Funktio palauttaa arvon TRUE, jos käyttäjätunnistin tunnistaa mobiiliverkkoselaimen tai jos laite hyväksyy WAP-sisältöä. Tunnistin tukee kaikkia moderneja mobiililaitteita, muun muassa Androidia, iPadia, iPhonea ja Windows Phonea.

IsMobile-funktion käyttö on hyödyllistä mediakyselyiden ja viewport-tagin rinnalla. Sillä voidaan antaa ehto, joka ottaa viewport-tagin käyttöön vain mobiililaitetta käytettäessä. Tähän ehtoon on myös lisätty käyttäjän asetusvalinta, jolloin käyttäjä voi itse päättää, haluaako hän käyttää Diariumin ajanavarauksen mobiilinäkymää.



Kuva 12. Diarium-mobiilikäyttöliittymän asetusvalinta.

```
<?php
$isMobile = $requestHandler->isMobile();
$asetukset = $this->Session->read('ajanvaraus.asetukset');

if($isMobile == 1 && $asetukset['ajanvaraus_mobiili'] == 1){
            echo '<meta name="viewport" content="initial-scale =
1.0">';
}
?>
```

Esimerkkikoodi 4. Viewport-tagi otetaan käyttöön vain mobiililaitteilla käyttäen CakePHP:n isMobile-funktiota ja mobiilikäyttöliittymän asetusvalintaa.

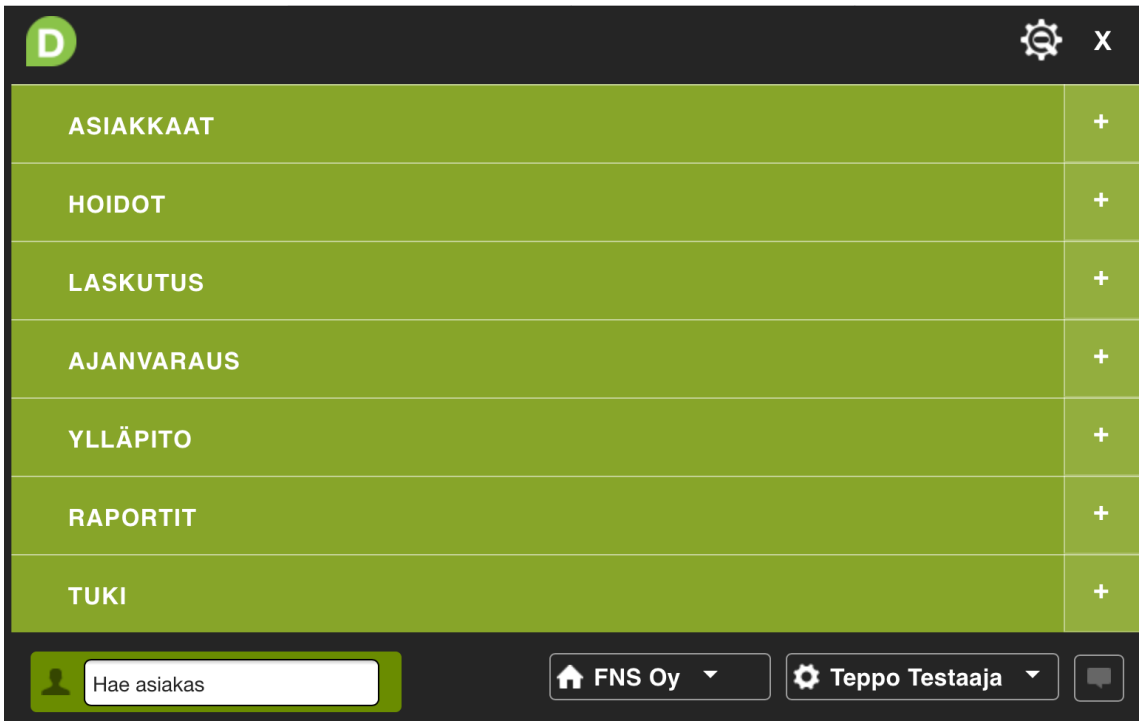
4.3.4 Mobiilivalikko

Diariumin päävalikko sisältää asiakkaanhakukentän, toimipiste- ja profiilinhallinta-alasvetovalikot, tiedotteet-painikkeen ja linkit näkymäryhmiin, joiden alla on linkit näkymiin. Päävalikko kokonaisuudessaan on tärkeä Diariumia käytettäessä, joten sen ominaisuuksien halutaan myös olevan toimivat mobiilinäkymässä. Alkuperäinen päävalikko on liian leveä alle 700 pikselin näytön leveydelle. Valikko jää ruudun ulkopuolelle, jos käytetään viewport-tagia skaalalla 1.0.

Kuva 13. Diariumin alkuperäinen päävalikko.

Toimivan sivunavigaation rakentaminen alusta asti mobiilialustoille on haastavaa. Kehittäjien avuksi on laaja valikoima erilaisia avoimen lähdekoodin mobiilivalikkoratkaisuja. FNS on käyttänyt aikaisemmissa projekteissaan toimivaksi todettua MeanMenu-jQuery-mobiilivalikkoa. Diariumin valikon näkymien linkkien HTML-koodi vastasi valmiiksi MeanMenun vaatimuksia, joten oli selvää, että mobiilivalikoksi tulee MeanMenu-liitännäinen. Päävalikon yläpalkin haun ja alasvetovalikoiden ominaisuudet halutaan pitää samanlaisena työpöytä- ja mobiiliversioissa. Tämä onnistuu piilottamalla mobiili.css (liite 2) tyylitiedostossa yläpalkki, kun mobiilivalikko on käytössä. Piilotus on oletusarvo, mutta yläpalkin saa näkyviin tarvittaessa erillistä nappia painamalla.

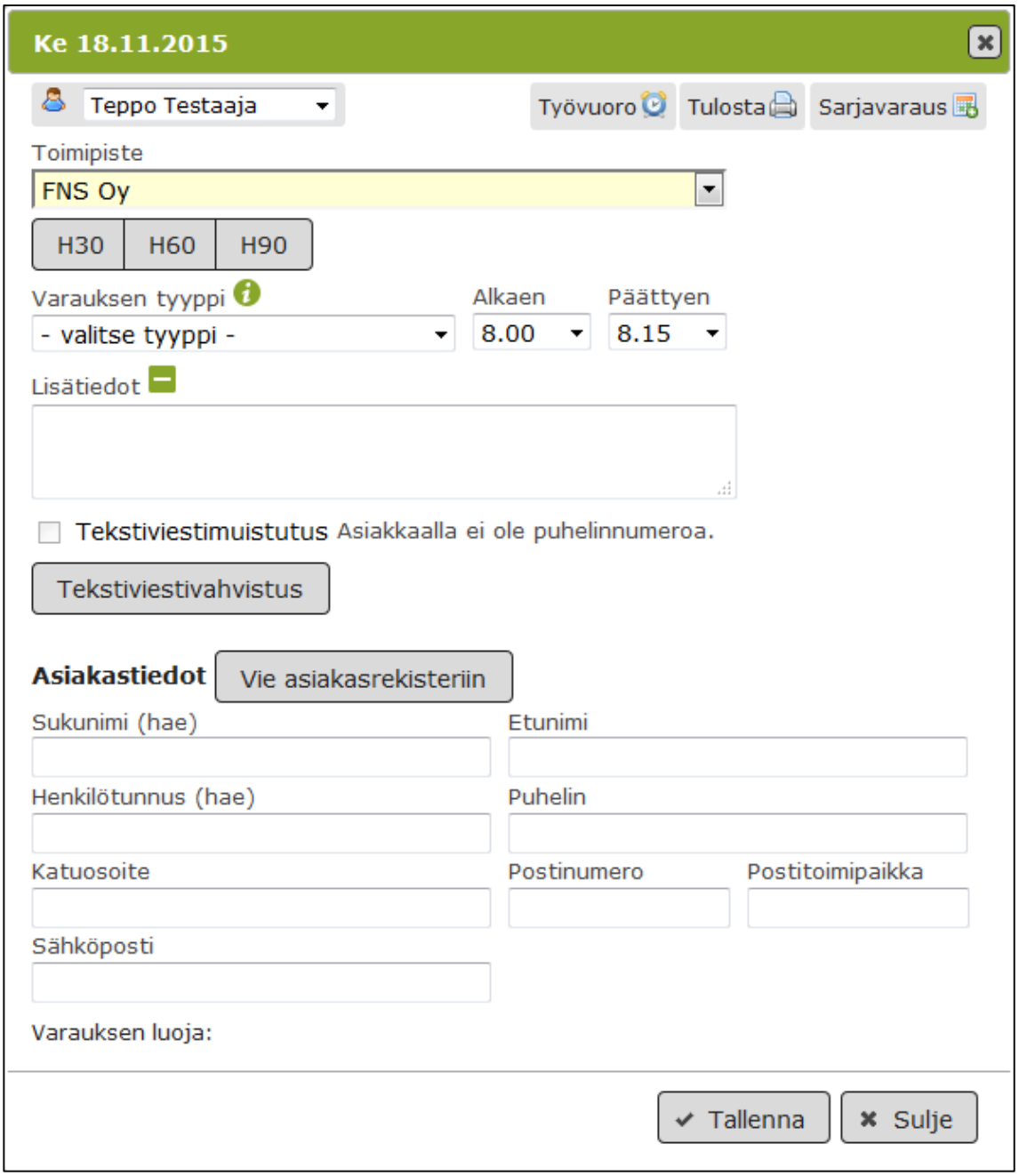
Kuva 14. Diariumin mobiilivalikko oletusasennossa.



Kuva 15. Diariumin mobiilivalikko molemmat alavalikot avattuna.

4.3.5 Bootstrap-modaali

Diarumin ajanvarausominaisuutta käytettäessä varauksen tiedot tallennetaan ponnahdusikkunaan. Tähän tarkoitukseen Diarumissa on käytössä jQueryn dialogiominaisuus. JQuery-dialogi toimii mobiililaitteilla, mutta ei ole lähtökohtaisesti mobiilioptimoitu. Tähän tarkoitukseen on monia mobiilioptimoituja ratkaisuja. Diariumin jatkokehityssuunnitelmaan kuuluu koko järjestelmän siirtäminen Bootstrap-työkalujen käyttöön. Looginen siirtymä on korvata jQuery-dialogit Bootstrap-modaaleiksi.



Kuva 16. Diariumin varauksen alkuperäinen jQuery-dialogi.

Bootstrap-modaali on ensisijaisesti mobiililaitteille suunniteltu kevyt JavaScript-ponnahdusikkuna. Se on jaettu kolmeen osaan: ylätunniste (header), sisältö (body) ja alatunniste (footer). JQuery.dialog-modal.js (liite 3) on JavaScript-funktio, joka generoi jQuery-dialogin ja sen sisällön Bootstrap-modaaliksi. Tämä funktio otetaan käyttöön, kun CakePHP:n isMobile-funktio tunnistaa päätelaitteen mobiililaitteeksi.

Kuva 17. Diariumin varauksen mobiilikäyttöliittymän Bootstrap-modaali.

4.3.6 Yhteenveto

Insinöörityön prototyyppisovellus toteutettiin lähes alkuperäisen suunnitelman mukaisesti, lukuun ottamatta muutamia suunniteltuja kehitysideoita. Tekniikat oli hyvin valittu, ja ne helpottivat projektin toteuttamista. Ajanvarausnäkymän mobiiliversio onnistuttiin pitämään ominaisuuksiltaan lähes yhtä toimivana kuin alkuperäinen ajanvarausnäkymä.

Kalenteri on nyt helppokäyttöisempi kosketusnäytöllä ja täyttää leveyssuunnassa koko mobiililaitteen näytön. Mobiilivalikko toimii sulavasti, ja sen käytettävyys on yhtä laaja kuin työpöytäversiossa. Varauksen modaali on mobiilioptimoitu ja toimii hyvin tablet-laitteilla.

Mobiilikäyttöliittymä toimii myös kohtalaisen hyvin älypuhelimilla. Tämä ominaisuus oli myös toivottu kehitysehdotus, mutta tablet-laitteilla on eniten mobiilikäyttäjiä, joten se oli kehittämisen lähtökohta. Älypuhelinten pienempi näytön koko vaatii vielä enemmän näkymän suunnittelua, jotta se toimisi puhelimen selaimessa täydellisesti.

Prototyyppi täyttää mobiilioptimoidun version kriteerit, mutta sitä jatkokehitetään asiakkaiden palautteen mukaisesti. Tätä prototyyppiä testataan Diarium-projektiryhmän sisäisesti, ja testien perusteella tehdään tarvittavat muutokset, jotta saadaan mobiilikäyttöliittymä tuotantoversioon.

Kuva 18. Diariumin ajanvarausjärjestelmä tietokoneella, tablet-laitteella ja älypuhelimella.

5 Pohdintaa

Insinöörityön tavoitteena oli tehdä ajanvarausjärjestelmän mobiilikehityssuunnitelma ja sen pohjalta kehittää prototyyppi. Työn tekemiseen meni aikaa noin kolme kuukautta. Ohjelmointitekniikoiden tutkiminen ja opetteleminen vei alussa paljon aikaa. Toteutuksen työkaluihin kannattaa kiinnittää huomiota, koska se minimoi työmäärän kehitysvaiheessa. Jos mobiiliversio tehdään järjestelmään jälkeenpäin, työ on laaja, koska näkymää joudutaan muuttamaan alkuperäisestä hyvin laajalti.

Projektiin sopivien viitekehyksien ja apukirjastojen käyttö mobiiliversioiden tekemisessä on hyödyllistä. Koodi on tehokkaampaa kehittäjälle, kun työkaluja käytetään ohjelmoidessa oikein. Myös mobiilijärjestelmän ylläpitäjän vaihtaminen on helpompaa, jos koodi on loogista ja laadukasta.

Prototyyppiä täytyy vielä kehittää ja testata laitekohtaisesti, jotta sen voisi viedä tuotantoon asti. Nyt se on toiminut testilaitteilla, mutta mobiilijärjestelmässä erittäin tärkeää on järjestelmäriippumattomuus, joka täytyy testata huolellisesti. Tulevaisuudessa kaikki Diariumin näkymät voisivat olla optimoituja mobiililaitteille.

Lopputulos oli kuitenkin tavoitteen mukainen. Aikaansaatu prototyyppi on moderni ja mobiilioptimoitu. Projektin edetessä ei vastaan tullut erityisiä ongelmia. Prototyyppiä kehitetään jatkossa, sillä se on todettu toimivaksi ratkaisuksi järjestelmään.

Lähteet

1 Yritys. 2015. Verkkodokumentti. Finnish Net Solutions Oy. <http://fns.fi/yritys> Luettu 26.11.2015.

2 Diarium. 2015. Verkkodokumentti. Finnish Net Solutions Oy. <http://www.diarium.fi> Luettu 13.4.2015.

3 Laaksonen, Antti, 2015. Ohjelmointiputka: Oppaat: PHP-ohjelmointi. Verkko-dokumentti. <http://www.ohjelmointiputka.net/oppaat/opas.php?tunnus=php_01> Luettu 22.11.2015.

4 Usage Statistics and Market Share of Server-side Programming Languages for Websites. 2015. Verkkodokumentti. W3Techs. <http://w3techs.com/technologies/overview/programming_language/all> Luettu 22.11.2015.

5 CakePHP Cookbook. 2015. Verkkodokumentti. Cake Software Foundation. <http://book.cakephp.org/3.0/_downloads/en/CakePHPCookbook.pdf> Luettu 13.4.2015.

6 CakePHP 3.0.0 is here. 2015. Verkkodokumentti. Cake Software Foundation. <http://bakery.cakephp.org/2015/03/22/CakePHP-3-0-0-is-Here.html> Luettu 22.11.2015.

7 CakePHP Cookbook. 2015. Verkkodokumentti. Cake Software Foundation. <http://book.cakephp.org/2.0/en/cakephp-overview/understanding-model-view-controller.html> Luettu 3.10.2015.

8 Duckett, Jon. 2014. Javascript & jQuery. Indianapolis: John Wiley & Sons.

9 JavaScript language resources. 2015. Verkkodokumentti. MDN. <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Language_Resources> Luettu 20.11.2015.

10 Choosing the Right JavaScript Framework for the Job. 2015. Verkkodokumentti. Lullabot. <https://www.lullabot.com/articles/choosing-the-right-javascript-framework-for-the-job> Luettu 3.10.2015.

11 Download jQuery. 2015. Verkkodokumentti. The jQuery Foundation. <https://jquery.com/download> Luettu 21.11.2015.

12 jQuery Usage Statistics. 2015. Verkkodokumentti. Builtwith. <http://trends.builtwith.com/javascript/jQuery> Luettu 21.11.2015.

13 Getting started. 2015. Verkkodokumentti. Bootstrap. <http://getbootstrap.com/getting-started > Luettu 22.11.2015

14 Ketteryys haltuun: Ketterän kehityksen yleiset periaatteet. 2015. Verkkodokumentti. Sininen Meteoriitti Oy. <https://www.meteoriitti.com/Artikkelisarjat/Ketteryys-haltuun/Ketteryys-haltuun-Ketteran-kehityksen-yleiset-periaatteet> Luettu 3.10.2015.

15 Ketterä kehitys. 2015. Verkkodokumentti. Technology Research Center. <http://trc.utu.fi/embedded/kasikirja/1/1> Luettu 22.11.2015.

16 MVC Architecture. 2015. Verkkodokumentti. Google Chrome. <https://developer.chrome.com/apps/app_frameworks> Luettu 22.11.2015.

17 Lehdonvirta, Pyry & Korpela, Jukka. 2013. HTML5 sovellusalustana. Helsinki: RPS-yhtiöt.

18 Variations on the Mobile Web. 2015. Verkkodokumentti. Google IO 2012. <http://smustalks.appspot.com/japan-12/#3> Luettu 3.10.2015.

19 Yahfoufi, Hussein. 2015. 3 Reasons to Choose Responsive Sites or Native Apps. Verkkodokumentti. <http://www.husseinyahfoufi.com/2014/02/responsive-sites-native-apps> Luettu 3.10.2015.

ajanvaraus.css

```
@CHARSET "ISO-8859-1";

@media only screen
and (min-device-width : 768px)
and (max-width: 2560px) {
  .responsive body, html{
          overflow-x: hidden;
  }
  .responsive #calendar{
          position: absolute;
          width: 98% !important;
  }
  .responsive h2{
          font-size: 150%;
  }
  .responsive #leftbar {
          display: none;
          font-size: 125%;
          max-width: 300px !important;
          padding: 10px 10px 10px 10px;
          border: 2px solid #dedddd;
          margin: 0px 0px 0px 0px !important;
          background: white;
          z-index: 10;
  }
  .responsive #leftbar input {
          margin-bottom: 15px;
  }
  .responsive #leftbar a{
          font-size: 100% !important;
  }
  .responsive #hoitajablokki {
          display: none;
  }
  .responsive #mobilemenu {
          display: block !important;
  }
  .responsive #header #asiakkaanValintaDiv {
          padding: 5px 15px 36px 36px;
  }
  .responsive #varausdialogi {
          font-size: 115% !important;
          top: 200px !important;
  }
  .responsive #alkaen {
          width: 85px !important;
  }
  .responsive #paattyen {
          width: 85px !important;
  }
  .responsive #varauksen_luoja {
          font-size: 100% !important;
  }
  .responsive .ui-autocomplete-input {
          height: 32px;
```

```
  }
  .responsive #varausdialogi .clearer {
          height: 10px;
  }
  .responsive #tallenna_asiakas {
          color: #0000EE;
  }
  .responsive .modal {
          top:10%;
  }
  .responsive input[type=radio] {
    border-radius: 50% !important;
  }
  .responsive #tyyppi_pika label {
    padding-right: 10px !important;
  }
}

@media (max-width: 767px) {
  .responsive body, html{
          overflow-x: hidden;
  }
  .responsive #calendar{
          position: absolute;
          width: 93% !important;
          margin: 0 auto;
  }
  .responsive h2{
          font-size: 150%;
  }
  .responsive #leftbar {
          display: none;
          font-size: 125%;
          max-width: 300px !important;
          padding: 10px 10px 10px 10px;
          border: 2px solid #dedddd;
          margin: 0px 0px 0px 0px !important;
          background: white;
          z-index: 10;
  }
  .responsive #leftbar input {
          margin-bottom: 15px;
  }
  .responsive #leftbar a{
          font-size: 100% !important;
  }
  .responsive #hoitajablokki {
          display: none;
  }
  .responsive #mobilemenu {
          display: block !important;
  }
  .responsive #header #asiakkaanValintaDiv {
          padding: 5px 15px 36px 36px;
  }
  .responsive #varausdialogi {
          font-size: 115% !important;
  }
  .responsive #alkaen {
```

```
        width: 85px !important;
}
.responsive #paattyen {
        width: 85px !important;
}
.responsive #varauksen_luoja {
        font-size: 100% !important;
}
.responsive .ui-autocomplete-input {
        height: 32px;
}
.responsive #varausdialogi .clearer {
        height: 10px;
}
.responsive #tallenna_asiakas {
        color: #0000EE;
}
.responsive .logo {
        display: none !important;
}
.responsive .fc-left {
        display: none;
}
.responsive .fc-center {
        display: none;
}
.responsive .login {
        float: left !important;
}
.responsive #user_id {
        width: 250px !important;
}
.responsive #corporate_id {
        width: 250px !important;
}
.responsive #lisatiedot {
        width: 250px !important;
}
.responsive #submenu {
        background-color: #000 !important;
}
.responsive #noteText {
        width: 250px !important;
}
.responsive #right{
        float: left !important;
}
.responsive #new_messages{
        float: left !important;
}
.responsive .modal {
        margin: 42px 0 0 0;
}
.responsive .fc-right {
        margin: 0 0 10px 0;
}
.responsive .mfp-content {
        top: -400px!important;
}
```

```
  .responsive #kalenteri-suodattimet {
          width: auto;
  }
  .responsive #ui-datepicker-div {
          top: 150px!important;
  }
  .responsive .btn-default {
          margin-top: 10px!important;
  }
  .responsive #tyyppi_pika label {
          padding-right: 10px !important;
  }
}
```

mobiili.css

```
@CHARSET "ISO-8859-1";

@media only screen
and (min-device-width : 768px)
and (max-width: 2560px) {
  .responsive #header {
          display: none;
          background-color: #252525 !important;
          background-image:none;
  }
  .responsive #tools {
          position: absolute;
          top: 7px!important;
          right: 50px!important;
  }
  .responsive #logo {
          display:none;
  }
  .responsive #logo2 {
          position: absolute;
          top: 7px!important;
          left: 15px!important;
  }
  .responsive #layout_asiakashaku {
          color: #252525;
  }
  .responsive .ui-autocomplete {
          overflow-x: hidden;
  }
  .responsive #asiakkaanValintaDiv{
          margin-left: 0px !important;
  }
  .responsive #content{
          padding-top: 10px;
  }
  .responsive #mainmenu li a {
          padding: 0px 37px 23px 10px;
  }
  .responsive #mobiili_asiakas {
          color: white;
          font-size: 18px;
          position: absolute;
          top: 14px!important;
          left: 65px!important;
          text-decoration: none;
  }
  .responsive .ui-autocomplete {
          position: absolute;
          top: 0;
          left: 0;
          cursor: default;
          z-index: 1060 !important;
  }
  .responsive .ui-autocomplete-input {
```

```
            position: relative;
    }
    .responsive div.tarvikehaku{
            z-index: 1052 !important;
    }
    .responsive div.maksutapa_dialog{
            z-index: 1053 !important;
    }
    .responsive div.maksutapa_dialog div.maksutapa_btn{
            border-style: solid;
            border-width: 1px;
            border-radius: 5px;
            padding-left: 25px;
    }
    .responsive .modal-header .close {
            font-size: 30px;
            margin-top: -5px !important;
    }
}

@media (max-width: 767px) {
    .responsive #selected_corporate {
            overflow: hidden;
            white-space: nowrap;
            text-overflow: ellipsis;
    }
    .responsive #header {
            display: none;
            background-color: #252525 !important;
            background-image:none;
    }
    .responsive #tools {
            position: absolute;
            top: 7px!important;
            right: 50px!important;
    }
    .responsive #logo2 {
            position: absolute;
            top: 7px!important;
            left: 15px!important;
    }
    .responsive #layout_asiakashaku {
            color: #252525;
    }
    .responsive .ui-autocomplete {
            overflow-x: hidden;

    }
    .responsive .left{
            float: left !important;
    }
    .responsive .right{
            float: left !important;
    }
    .responsive #asiakkaanValintaDiv{
            margin-left: 20px !important;
    }
    .responsive #content{
            padding-top: 10px;
```

```
  }
  *.responsive #mainmenu li a {
          padding: 0px 37px 23px 10px;
  }
  .responsive #mobiili_asiakas {
          color: white;
          font-size: 18px;
          position: absolute;
          top: 14px!important;
          left: 65px!important;
          text-decoration: none;
          width: 170px;
          overflow-x: hidden;
          white-space: nowrap;
          text-overflow: ellipsis;
  }
  .responsive .ui-autocomplete {
          position: absolute;
          top: 0;
          left: 0;
          cursor: default;
          z-index: 1060 !important;
  }
  .responsive .ui-autocomplete-input {
          position: relative;
  }
  .responsive div.tarvikehaku{
          z-index: 1052 !important;
  }
  .responsive div.maksutapa_dialog{
          z-index: 1053 !important;
  }
  .responsive div.maksutapa_dialog div.maksutapa_btn{
          border-style: solid;
          border-width: 1px;
          border-radius: 5px;
          padding-left: 25px;
  }
  .responsive .modal-header .close {
          font-size: 30px;
          margin-top: -5px !important;
  }
}
```

jquery.dialog-modal.js

```
;(function($) {

  $.fn.dialog = function(options) {

    var self    = this
      , $this   = $(self)
      , $body   = $(document.body)
      , $msgbox = $this.closest('.dialog')
      , parentDataName = 'dialog-parent'
      , arg1    = arguments[1]
      , arg2    = arguments[2]
      ;

    var create = function() {
           var title = $this.attr('class');

           console.log(title );

      var msghtml
        = ''
        + '<div class="dialog modal '+title+'">'
        + '<div class="modal-dialog">'
        +   '<div class="modal-content">'
        +     '<div class="modal-header">'
        +         '<button type="button"
class="close">&times;</button>'
        +         '<h4 class="modal-title"></h4>'
        +     '</div>'
        +     '<div class="modal-body"></div>'
        +     '<div class="modal-footer"></div>'
        +   '</div>'
        + '</div>'
        + '</div>'
        ;


      $msgbox = $(msghtml);
      $(document.body).append($msgbox);
      $msgbox.find(".modal-body").append($this);
    };

    var createButton = function(_options) {
      var buttons = (_options || options || {}).buttons || {}
        , $btnrow = $msgbox.find(".modal-footer");

      //clear old buttons
      $btnrow.empty();

      var isButtonArr = buttons.constructor == Array;

      for (var button in buttons) {
        var btnObj  = buttons[button]
          , id      = ""
          , text    = ""
```

```
          , classed = "btn-default"
          , click   = "";

        if (btnObj.constructor == Object) {
          id      = btnObj.id;
          text    = btnObj.text;
          classed = btnObj['class'] || btnObj.classed || classed;
          click   = btnObj.click;
        }

        //Buttons should be an object, etc: { 'close': function { } }
        else if (!isButtonArr && btnObj.constructor == Function) {
          text  = button;
          click = btnObj;
        }

        else {
          continue;
        }

        //<button data-bb-handler="danger" type="button" class="btn
btn-danger">Danger!</button>
        $button = $('<button type="button"
class="btn">').addClass(classed).html(text);

        id && $button.attr("id", id);
        if (click) {
          (function(click) {
            $button.click(function() {
              click.call(self);
            });
          })(click);
        }

        $btnrow.append($button);
      }

      $btnrow.data('buttons', buttons);
    };

    var show = function() {
      // call the bootstrap modal to handle the show events (fade ef-
fects, body class and backdrop div)
      $msgbox.modal('show');
    };

    var close = function(destroy) {
      // call the bootstrap modal to handle the hide events and remove
msgbox after the modal is hidden
      $msgbox.modal('hide').one('hidden.bs.modal', function() {
          if (destroy) {
              $this.data(parentDataName).append($this);
              $msgbox.remove();
          }
      });
    };

    if (options.constructor == Object) {
```

```
      !$this.data(parentDataName) && $this.data(parentDataName,
$this.parent());

      if ($msgbox.size() < 1) {
        create();
      }
      createButton();
      $(".modal-title",  $msgbox).html(options.title || "");
      $(".modal-dialog", $msgbox).addClass(options.dialogClass || "");
      $(".modal-header .close", $msgbox).click(function() {
        var closeHandler = options.onClose || close;
        closeHandler.call(self);
      });
      (options['class'] || options.classed) &&
$msgbox.addClass(options['class'] || options.classed);
      options.autoOpen !== false && show();
    }

    if (options == "destroy") {
      close(true);
    }

    if (options == "close") {
      close();
    }

    if (options == "open") {
      show();
    }

    if (options == "option") {
      if (arg1 == 'buttons') {
        if (arg2) {
          createButton({ buttons: arg2 });
          show();
        } else {
          return $msgbox.find(".modal-footer").data('buttons');
        }
      }
    }

    return self;
  };

})(jQuery);
```